CNN-based Prediction for Lossless Coding of Photographic Images

Ionut Schiopu, Yu Liu and Adrian Munteanu
Department of Electronics and Informatics (ETRO)
Vrije Universiteit Brussel (VUB)
Brussels, BELGIUM
{ischiopu, yliub, acmuntea}@etrovub.be

***Abstract*—The paper proposes a novel prediction paradigm in image coding based on Convolutional Neural Networks (CNN). A deep neural network is designed to provide accurate pixel-wise prediction based on a causal neighbourhood. The proposed CNN prediction method is trained on the high-activity areas in the image and it is incorporated in a lossless compression system for high-resolution photographic images. The system uses the proposed CNN-based prediction paradigm as well as LOCO-I, whereby the predictor selection is performed using a local entropy-based descriptor. The prediction errors are encoded using a CALIC-based reference codec. The experimental results show a good performance for the proposed prediction scheme compared to state-of-the-art predictors. To our knowledge, the paper is the first to introduce CNN-based prediction in image coding, and demonstrates the potential offered by machine learning methods in coding applications.**

I. Introduction

The technological advances in camera sensors offer the possibility to capture high resolution images, e.g., the 4K Ultra High Definition (UHD) resolution of 3840×2160 pixels or higher. In photographic imagery, users are often interested in processing the original information acquired by the camera sensors. Due to the spatial resolutions and resulting data rates, highly-efficient lossless compression algorithms are essential when compressing this type of images.

Traditional methods for lossless image compression follow a pixel-wise predictive coding paradigm. In these methods the value of the current pixel is predicted based on a causal neighbourhood; the prediction error is subsequently processed (e.g., context modeling, remapping), and the modelled error is encoded using a variable-length entropy coding algorithm.

JPEG-LS [1] is one of the most popular lossless compression methods from this category, implementing the well-known LOCO-I predictor. JPEG-LS achieves low complexity by using a simple causal neighbourhood of only three pixels, and using Golomb-like codes in the entropy coding stage.

Context-based Adaptive Lossless Image Coder (CALIC) [2] is one of the best solutions for lossless image compression. CALIC computes an edge descriptor, based on a causal neighbourhood of six pixels, which is used to distinguish seven types of edges based on strength (from weak to sharp) and edge direction (horizontal or vertical). CALIC designs an appropriate predictor for each edge type, processes the prediction error using a complex context modeling procedure, and incorporates an efficient context-adaptive variable-length entropy coding algorithm.

Existing state-of-the-art lossless compression algorithms were devised for images having a much lower spatial resolution than the resolutions encountered today. The employed prediction mechanisms are not able to provide an efficient prediction in regions with strong edges and highly textural areas, abundantly present in high-resolution imagery such as UHD and beyond. Such regions contain strong gradients and a lot of nonlinearities; in such areas, traditional predictors are not able to accurately estimate the strength of the sharp edges.

In this paper, we propose a novel CNN-based prediction paradigm, which is specifically designed to predict highly textural regions in high-resolution images. In this sense, the paper explores the potential offered for image coding by Convolutional Neural Networks [3], [4]. One notes that CNNs have recently demonstrated state-of-the-art performance in many other domains, such as inpainting, super-resolution, denoising, to name a few.

Machine learning tools for image compression have been recently explored in [5], offering a solution for block-based lossy image compression; the architecture of [5] consists of a Recurrent Neural Network (RNN) based encoder and decoder operating on image blocks, a binarizer, and a neural network for entropy coding. Although promising, the coding performance of [5] does not exceed that of conventional JPEG. Another auto-encoder block-based lossy coding solution is presented in [6], the algorithm offering natural scalability while visually outperforming JPEG and JPEG-2000 codecs.

Although promising, these methods are not lossless codecs. Furthermore, they do not devise *predictors* for image coding, but treat the employed machine learning tools as block classifiers and block encoders. In this paper, we treat machine learning tools completely differently: we do not regard them as block encoders, but we employ them as tools for pixel prediction, competing with conventional prediction tools and being part of conventional image coding machinery.

In summary, the paper's contributions are as follows: **(a)** a novel pixel-wise CNN-based predictor specialized in predicting sharp edges; **(b)** a novel prediction scheme for lossless compression of 4K UHD photographic images; **(c)** a local entropy-based descriptor partitioning the image into regions containing weak or sharp edges, and enabling automatic

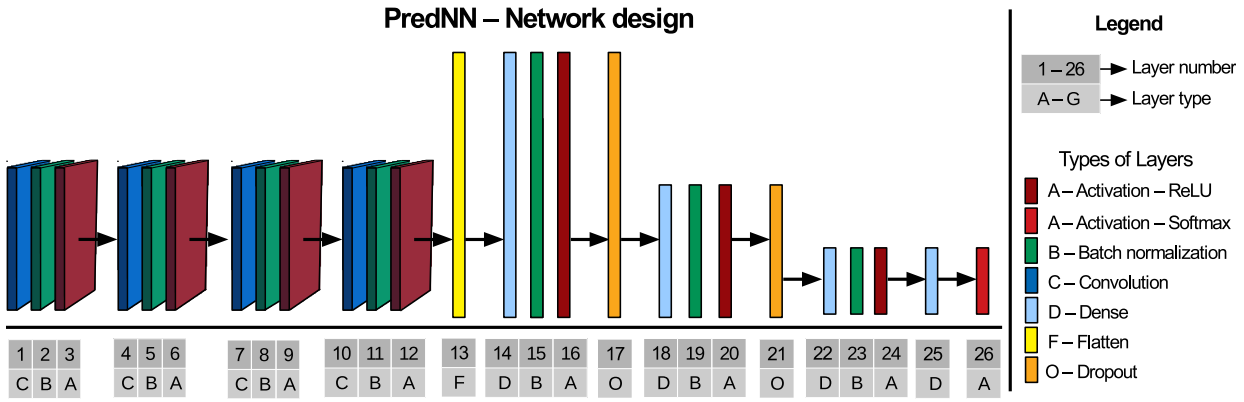

Fig. 1: The deep Predictive Neural Network (PredNN) is form of 26 layers of the following core layer types: *(A)* Activation; *(B)* Batch normalization; *(C)* Convolution; *(D)* Dense; *(F)* Flatten; *(O)* Dropout.

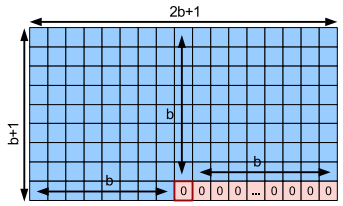

Fig. 2: The causal neighbourhood $\mathcal{N}(b)$ of size $(b+1) \times (2b+1)$ selects pixels up to the distance b from the current position. The last $b+1$ values on the last row are set to 0.

predictor selection at pixel level; **(d)** a CALIC-based entropy codec enabling the comparison of the different predictors.

The remainder of the paper is organized as follows. Section II describes the proposed CNN prediction method. Section III describes the prediction scheme. Section IV reports the experimental evaluation, and Section V draws the conclusions.

II. CNN Predictor

In this paper, we propose a CNN architecture for pixel-wise prediction. The predictor, dubbed Predictive Neural Network (PredNN), computes the CNN prediction, denoted as $\hat{I}_{CNN}(x,y)$, for the current pixel position (x,y), in the input image I, based on a causal neighbourhood of this pixel. The proposed CNN architecture is depicted in Figure 1. The CNN prediction error, $\varepsilon_{CNN}(x,y)$, is computed as:

$$\varepsilon_{CNN}(x,y) = I(x,y) - \hat{I}_{CNN}(x,y). \quad (1)$$

As depicted in Figure 2, PredNN's input is the rectangular-shaped causal neighbourhood, $\mathcal{N}(b)$, of the current pixel position, $(b+1, b+1)$, which selects the pixels found at a maximum distance b (horizontally and vertically) from $(b+1, b+1)$. $\mathcal{N}(b)$ is stored in a matrix of size $(b+1) \times (2b+1)$, where the last $b+1$ values in the last row are set to 0 - see Figure 2.

The deep network of PredNN is using a forward propagation model and contains 26 layers of the following types:

(A) *Activation*: uses one of the following activation functions: *(i)* a rectified linear unit ReLU activation function [7], defined as $f(z) = max(0,z)$, is used after a batch normalization layer in the layers: $3, 6, 9, 12, 16, 20, 24$; *(ii)* a Softmax (or normalized exponential) function [8] is used in the last layer; it is defined as: $\sigma(\mathbf{z}) = \frac{e^{z_j}}{\sum_{k=1}^{K} e^{z_k}}, j = 1:K$.

(B) *Batch normalization*: is used after a convolution or a dense layer in the following layers: $2, 5, 8, 11, 15, 19, 23$.

(C) *Convolution*: a 3×3 convolution is used in the first layer, and a 2×2 convolution is used in the following layers: $4, 7, 10$, all having 128 output filters.

(D) *Dense*: is used after a flatten or dropout layer in the layers $14, 18, 22, 25$, and contains $2^{10}, 2^9, 2^8$, and respectively the number of output classes (e.g., 2^8 for an 8-bit image).

(F) *Flatten*: is vectorizing the output in the layer number 13.

(O) *Dropout*: is randomly setting to 0 the fraction rate $f_D = 0.5$ of the input at each update during training time; it is used in layers 17 and 21 to prevent network overfitting.

Figure 1 shows that PredNN is designed to first process the input image with several convolution layers to recognize the local patterns and to identify a feature vector, and then to process the feature vector with several dense layers in a way inspired by the traditional predictors to compute the prediction. In the following convolution/dense layers: $1, 4, 18, 22, 25$, the kernel initialization is set as the He [9] uniform variance scaling initializer, where the samples are drawn from a uniform distribution within $[-\alpha, \alpha], \alpha = \sqrt{\frac{6}{n_1}}$, where n_1 is the number of input units in the weight tensor. The remaining convolution and dense layers use the Xavier [10] uniform initializer, where the samples are drawn from a uniform distribution within $[-\alpha, \alpha], \alpha = \sqrt{\frac{6}{n_1+n_2}}$, where n_2 is the number of output units in the weight tensor. The kernel regularizer in the convolution and dense layers is done using the ℓ_2 weight regularization penalty, also known as Ridge regression [11].

The Adaptive Momentum Estimation (Adam) [12] optimization algorithm is deployed for the model. In summary, the method stores the exponentially decaying average of the past squared gradients, $v_t = \beta_1 m_{t-1} + (1-\beta_1) g_t$ (first momentum),

and of the past gradients, $m_t = \beta_2 v_{t-1} + (1-\beta_2) g_t^2$ (second momentum). The Adam update rule is

$$\theta_{t+1} = \theta_t - \frac{\eta}{\sqrt{\hat{v}_t} + \epsilon} \hat{m}_t, \tag{2}$$

where $\hat{m}_t = \frac{m_t}{1-\beta_1^t}$ and $\hat{v}_t = \frac{v_t}{1-\beta_2^t}$ are the bias-corrected momentum estimates, η is the learning rate, and $(\beta_1, \beta_2, \epsilon)$ are the Adam parameters. More details can be found in [12]. The categorical cross-entropy function is used as the model's loss function.

III. Proposed Prediction Scheme

In this paper, we propose a novel prediction scheme, whereby the highly textural regions containing sharp edges are predicted using PredNN, while the flat areas containing weak edges are predicted using the traditional LOCO-I predictor. The prediction errors are encoded using a CALIC-based reference coder which enables the experimental assessment of the various prediction methods.

The LOCO-I and CALIC predictors are summarized in Section III-A. The novel prediction scheme is described using an entropy-based descriptor, see Section III-B. The entropy coding system is detailed in Section III-C.

A. State of the art Predictors

For current pixel position, (x, y), let us denote: $n = I(x-1, y), w = I(x, y-1), nw = I(x-1, y-1), ne = I(x-1, y+1), ww = I(x, y-2), nn = I(x-2, y), nne = I(x-2, y+1)$. The prediction schemes of the two most important state-of-the-art predictors are summarized as follows:

1) LOCO-I: This predictor [1] is specialized in predicting the image's flat areas, being defined as follows:

$$\hat{I}_{LOCO}(x,y) = \begin{cases} \min(n,w) & \text{if } ne \geq \max(n,w), \\ \max(n,w) & \text{if } ne \leq \min(n,w), \\ n + w - ne & \text{otherwise,} \end{cases} \tag{3}$$

where, the LOCO-I prediction error is computed as

$$\varepsilon_{LOCO}(x,y) = I(x,y) - \hat{I}_{LOCO}(x,y). \tag{4}$$

2) CALIC: A local edge descriptor is introduced in [2] as $d = d_v - d_h$, where $d_h = |w - ww| + |n - nw| + |n - ne|, d_v = |w - nw| + |n - nn| + |ne - nne|$, and is used to define seven prediction areas. The CALIC prediction is computed as:

$$\hat{I}_{CALIC}(x,y) = \begin{cases} n & \text{if } d < -80, \\ \frac{\hat{I}_c + n}{2} & \text{if } -80 \leq d < -32, \\ \frac{3\hat{I}_c + w}{2} & \text{if } -32 \leq d < -8, \\ \hat{I}_c & \text{if } -8 \leq d \leq 8, \\ \frac{3\hat{I}_c + w}{4} & \text{if } 8 < d \leq 32, \\ \frac{\hat{I}_c + w}{2} & \text{if } 32 < d \leq 80, \\ w & \text{if } d > 80, \end{cases} \tag{5}$$

where $\hat{I}_c = \frac{n+w}{2} + \frac{ne-nw}{4}$. The CALIC prediction error is

$$\varepsilon_{CALIC}(x,y) = I(x,y) - \hat{I}_{CALIC}(x,y). \tag{6}$$

B. Entropy-based Descriptor

In this section, we introduce a novel image descriptor, which is obtained by thresholding the image's local entropy. The descriptor outputs the binary decision used for computing the prediction error either by using PredNN or LOCO-I.

The two-steps algorithm is summarized as follows: *(i)* evaluate the pixel's local activity using a causal neighbourhood; *(ii)* use an adaptive threshold procedure for partitioning the image into regions containing weak or sharp edges. The local activity is measured using the information entropy, $H(\mathbf{p}) = -\sum_{i=1}^{N} p_i \log p_i$, where the vector of probabilities $\mathbf{p}(\mathcal{N}(b_H)(x,y)) = [p_1 \quad p_2 \quad \dots \quad p_N]$ is computed using the distribution of symbols in a causal neighbourhood $\mathcal{N}(b_H)$ around the current pixel position, (x, y), of size $(b_H + 1) \times (2b_H + 1)$ pixels, where b_H is the descriptor's neighbourhood parameter. The local entropy matrix, $\mathbf{J}_H$, is defined as:

$$J_H(x,y) = H(\mathbf{p}(\mathcal{N}(b_H)(x,y))). \tag{7}$$

The descriptor matrix, denoted $\mathbf{D}_H$, is set as follows:

$$D_H(x,y) = \begin{cases} 0, & \text{use LOCO-I if } J_H(x,y) \leq T_D; \\ 1, & \text{use PredNN if } J_H(x,y) > T_D, \end{cases} \tag{8}$$

where T_D is the adaptive threshold defined as:

$$T_D = T + \Delta T, \tag{9}$$

where T is the imposed threshold, and $\Delta T = \alpha(T - \bar{m}_J)$ is the adaptation step, where $\bar{m}_J$ is the mean-value in $\mathbf{J}_H$. On one hand, if the image contains a lot of high-activity areas, then $\bar{m}_J > T$ and $\Delta T < 0$, so that T_D is decreased and most of the pixels are predicted using the PredNN predictor, which is especially designed for this type of areas. On the other hand, if the image contains a lot of flat areas, then $\bar{m}_J < T$ and $\Delta T > 0$, so that T_D is increased and most of the pixels are predicted using the LOCO-I predictor. Note that $\bar{m}_J$ is encoded together with the descriptor's parameters, α and T.

C. Entropy Coder

An image codec commonly integrates with its system architecture a good prediction scheme, a specific context modeling algorithm, and an entropy coder, which all together are contributing to improve the codec's compression performance.

In this paper, the performance of the proposed prediction scheme is tested using a CALIC-based reference codec, which implements the most important features from CALIC [2]:

1) Binary mode: A coding mechanism for encoding (in a "brute-force" manner) the image's flat areas detected using a six-pixels neighbourhood in a sequential codec.

2) Context adaptive error modeling: The algorithm has the following main steps: *(i)* Adjust the prediction in each prediction context by imposing that the context's mean is always converging to zero. The context is defined by the energy of the local binary pattern and the errors in the local neighbourhood. *(ii)* Predict the error sign using context modeling. *(iii)* Remap the modeled error to $[0,\ 2^B - 1]$ range, so that there is no need to encode the error sign.

TABLE I: Lossless Compression Results in *bpp*

Predictor		CALIC	LOCO-I	PredNN	LocoNN
TEST set	Avg.	2.973	2.927	2.910	**2.868**
TRAINING set	Avg.	3.507	3.470	3.419	**3.384**

3) Variable-Length Entropy Coding: An entropy coding algorithm built on the concept of histogram tail truncation, where in each region the symbols are encoded using an Adaptive Markov Model and an escape mode is introduced so that the model is changed if the error exceeds the model's coding symbol range.

IV. EXPERIMENTAL EVALUATION

A. Experimental Setup

The 4K UHD images used in our tests were collected from [13], by randomly selecting a number of 68 grayscale images with resolution of 3840×2160, and were divided into Training set (8 images) and Test set (60 images). To test the concept of pixel-wise CNN prediction, we selected the images having at least one of the following image tags: natural images, mountains, forest, animals, city lights, panoramic views.

The following algorithm configuration was used in our tests: $b = 15$, resulting $\mathcal{N}(15)$ as the input images of size 16×31; $b_H = 8$, resulting in a neighbourhood $\mathcal{N}(8)$ of size 9×17 used to compute the local-entropy; $\alpha = 1, T = 3.5$, resulting $T_D = 7 - \bar{m}_J$ as the threshold of the descriptor; $\beta_1 = 0.9, \beta_2 = 0.999, \epsilon = 10^{-8}$ are the parameters of the Adam optimizer. The algorithm was tested with input images of different sizes and $b = 15$ was found to yield the best performance. T_D was set after comparing the performance of the predictors using the entropy as a metric, see Figure 3. For the α parameter we recommend to use values within the range $[0, 1]$.

PredNN was developed using the Keras API, while the CALIC-based reference codec was implemented in C++. The network was trained using 10^6 training patches (of size 16×31) selected from the Training set as follows: for each image, randomly select $125 \cdot 10^3$ training patches out of the approximately $8.29 \cdot 10^6$ image patches ($3840 \cdot 2160$), i.e., only 1.5% of the image data in the Training set is used for training. PredNN was trained during a number of 50 epochs, using a batch size of 2000, $\eta = 10^{-4}$, and using a 90%-10% ratio for splitting the 10^6 training patches into training-validation data.

B. Experimental results

Here, we compare the performance of the following prediction schemes: LOCO-I [1]; CALIC [2]; the proposed CNN-based predictor alone, PredNN, and integrated together with LOCO-I in the proposed prediction scheme, called LocoNN.

Figure 3(a) shows the performance comparison of the four prediction schemes for the set of 4K UHD images. One can notice that PredNN obtains good results for the high-entropy images, while LOCO-I obtains good results for the low-entropy images. However, by integrating the two predictors, LocoNN obtains the best results for almost all images. The threshold $T = 3.5$ bits per pixel (*bpp*) can be used for separating the low and high entropy images.

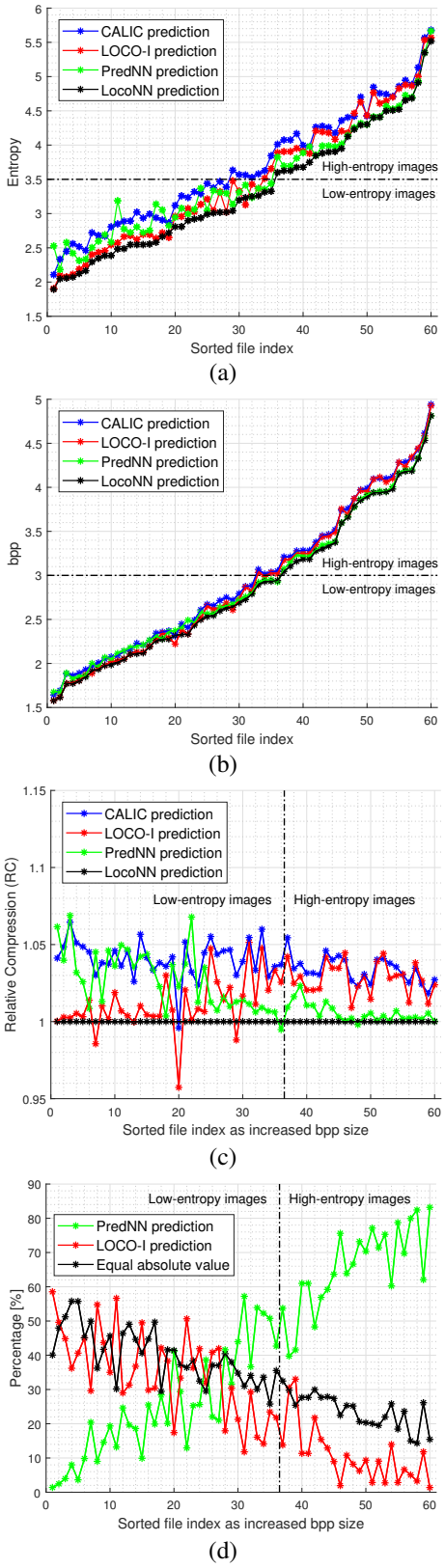

Fig. 3: **(a)** Entropy results. **(b)** Compression results in *bpp*. **(c)** Compression results in *RC*. **(d)** The weights of PredNN and LOCO-I in LocoNN.

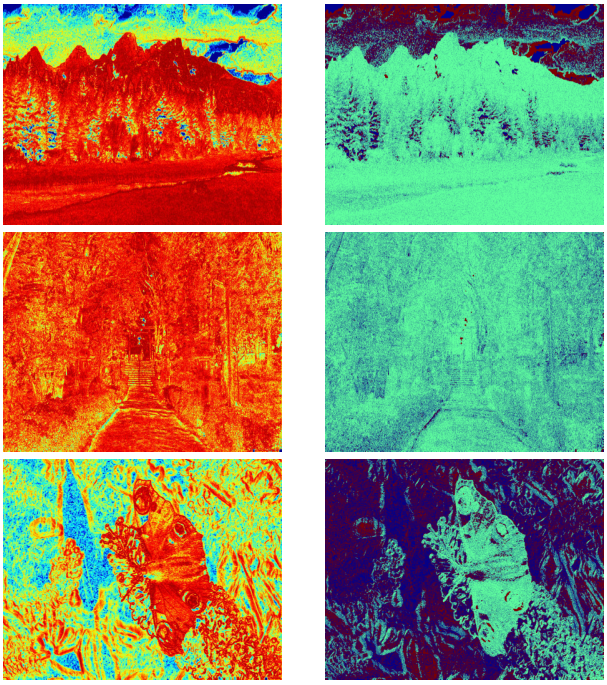

Fig. 4: **(Left)** Jet colormap coding of the local entropy: *(blue)* small local entropy; *(red)* large local entropy. **(Right)** LocoNN Predictor Composition: *(green)* PredNN prediction; *(red)* LOCO-I prediction; *(blue)* equal absolute value.

Figure 3(b) shows the compression results in *bpp* obtained by running the CALIC-based reference codec and by using in turn each of the four prediction schemes. Table I shows the average results for each set, where one can notice that LocoNN obtains a 3.66% improvement compared to CALIC and 2.06% compared to LOCO-I. Figure 3(c) show the Relative Compression (RC) results of the four prediction schemes relative to LocoNN, while keeping the same image order as in Figure 3(b). One can notice that LocoNN obtains the best results for $55/60$ images in the Test set, and PredNN has a similar performance to CALIC for low-entropy images, and a good performance for the high-entropy images. The tests have shown that $T_r = T - \bar{m}_\Delta$ can be used for separating the low and high entropy images, where $\bar{m}_\Delta$ is the mean value of the difference between the entropy and the corresponding compression result of the CALIC-based reference codec, for each image. In this paper, we found $\bar{m}_\Delta = 0.5$ *bpp* and obtained $T_r = 3$ *bpp*. An interesting observation is that, in general, LOCO-I has a better performance than CALIC, despite the fact that a CALIC-based reference codec is used for which the CALIC predictor is expected to have some advantage over the other prediction schemes.

Figure 4 shows examples of four pairs of color-coded images of $\mathbf{J}_H$ and of the pixel positions where the PredNN and LOCO-I predictors are used. Figure 3(d) shows the weight percentage of the two predictor in LocoNN, while keeping the image order as in Figure 3(b). For the high-entropy images, the tests have shown that PredNN has a mean weight of 65% in LocoNN, without counting the equal absolute value cases. Moreover, the proposed descriptor integrates efficiently the two predictors into the LocoNN prediction scheme.

V. Conclusions

In this paper we introduce a novel pixel-wise CNN-based predictor for lossless compression of 4K UHD photographic images. A CNN architecture was proposed and the network was trained to have a good performance for the demanding task of predicting the high activity areas in high-entropy images. PredNN and LOCO-I were efficiently integrated in the LocoNN prediction scheme and a novel local-entropy based descriptor was used to select between the two predictors.

The experimental results show that PredNN outperforms the traditional predictors in high-activity areas and has a high contribution in the composition of LocoNN. In the future, we plan to modify the structure of PredNN's network and to train it with a much larger dataset, which will contain all region types, not only the sharp edges. This is expected to generalize the applicability of CNN-based prediction to any region types.

To our knowledge, the paper is the first to introduce CNN-based prediction in image coding and proves the potential offered by modern machine learning tools in data compression.

Acknowledgement

The work in this paper has been supported by the 3DLicorneA project funded by the Brussels Institute for Research and Innovation (Innoviris).

References

[1] M. Weinberger, G. Seroussi, and G. Sapiro, "The LOCO-I lossless image compression algorithm: principles and standardization into JPEG-LS," *IEEE Trans. Image Process.*, vol. 9, no. 8, pp. 1309–1324, Aug. 2000.

[2] X. Wu and N. Memon, "Context-based, adaptive, lossless image coding," *IEEE Trans. Commun.*, vol. 45, no. 4, pp. 437–444, Apr. 1997.

[3] Y. LeCun, B. Boser, J. Denker, D. Henderson, R. Howard, W. Hubbard, and L. Jackel, "Backpropagation applied to handwritten zip code recognition," *Neural Computation*, vol. 1, no. 4, pp. 541–551, Dec. 1989.

[4] A. Krizhevsky, I. Sutskever, and G. Hinton, "Imagenet classification with deep convolutional neural networks," in *Proc. Advances in Neural Information Processing Systems*, 2012.

[5] G. Toderici, D. Vincent, N. Johnston, S. Hwang, D. Minnen, J. Shor, and M. Covell, "Full resolution image compression with recurrent neural networks," in *Proc. Conference on Computer Vision and Pattern Recognition (CVPR)*, Jul. 2017, pp. 5435–5443.

[6] K. Gregor, F. Besse, D. Jimenez Rezende, I. Danihelka, and D. Wierstra, "Towards Conceptual Compression," *ArXiv e-prints*, Apr. 2016.

[7] V. Nair and G. Hinton, "Rectified linear units improve restricted boltzmann machines," in *Proc. Int. Conf. Machine Learning (ICML)*, Washington, DC, USA, 2010, pp. 807–814.

[8] C. Bishop, *Pattern Recognition and Machine Learning*. Secaucus, NJ, USA: Springer-Verlag New York, Inc., 2006.

[9] K. He, X. Zhang, S. Ren, and J. Sun, "Delving Deep into Rectifiers: Surpassing Human-Level Performance on ImageNet Classification," *ArXiv e-prints*, Feb. 2015.

[10] X. Glorot and Y. Bengio, "Understanding the difficulty of training deep feedforward neural networks," in *Proc. Int. Conf. Artificial Intell. Statist.*, vol. 9, Chia Laguna Resort, Sardinia, Italy, May 2010, pp. 249–256.

[11] A. Hoerl and R. Kennard, "Ridge regression: Biased estimation for nonorthogonal problems," *Technometrics*, vol. 42, no. 1, pp. 80–86, Feb. 2000.

[12] D. Kingma and J. Ba, "Adam: A Method for Stochastic Optimization," *ArXiv e-prints*, Dec. 2014.

[13] "4K UHD Photographic Images," http://www.ultrahdwallpapers.net/nature, accessed: 2017-08-25.